Chapter 1
Basic Structure of Computers

1.1.
- Transfer the contents of register PC to register MAR
- Issue a Read command to memory, and then wait until it has transferred the requested word into register MDR
- Transfer the instruction from MDR into IR and decode it
- Transfer the address LOCA from IR to MAR
- Issue a Read command and wait until MDR is loaded
- Transfer contents of MDR to the ALU
- Transfer contents of R0 to the ALU
- Perform addition of the two operands in the ALU and transfer result into R0
- Transfer contents of PC to ALU
- Add 1 to operand in ALU and transfer incremented address to PC

1.2.
- First three steps are the same as in Problem 1.1
- Transfer contents of R1 and R2 to the ALU
- Perform addition of two operands in the ALU and transfer answer into R3
- Last two steps are the same as in Problem 1.1

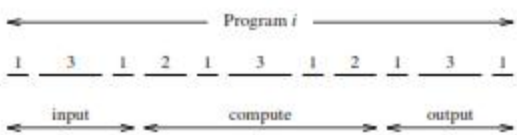
1.3. (*a*)

```
Load   A,R0
Load   B,R1
Add    R0,R1
Store  R1,C
```

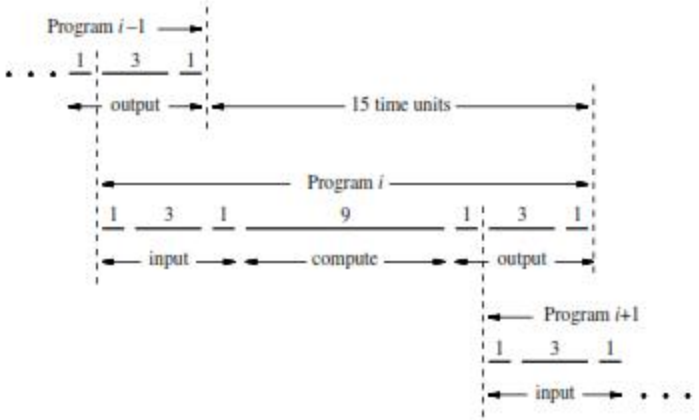
(*b*) Yes;

```
Move   B,C
Add    A,C
```

1.4. (*a*) Non-overlapped time for Program *i* is 19 time units composed as:



Overlapped time is composed as:



Time between successive program completions in the overlapped case is 15 time units, while in the non-overlapped case it is 19 time units.

Therefore, the ratio is 15/19.

(*b*) In the discussion in Section 1.5, the overlap was only between input and output of two successive tasks. If it is possible to do output from job $i-1$, compute for job i, and input to job $i+1$ at the same time, involving all three units of printer, processor, and disk continuously, then potentially the ratio could be reduced toward 1/3. The OS routines needed to coordinate multiple unit activity cannot be fully overlapped with other activity because they use the processor. Therefore, the ratio cannot actually be reduced to 1/3.

1.5. (*a*) Let $T_R = (N_R \times S_R) / R_R$ and $T_C = (N_C \times S_C) / R_C$ be execution times on the RISC and CISC processors, respectively. Equating execution times and clock rates, we have

$$1.2\,N_R = 1.5\,N_C$$

Then

$$N_C / N_R = 1.2 / 1.5 = 0.8$$

Therefore, the largest allowable value for N_C is 80% of N_R.

(*b*) In this case

$$1.2\,N_R / 1.15 = 1.5\,N_C / 1.00$$

Then

$$N_C / N_R = 1.2 / (1.15 \times 1.5) = 0.696$$

Therefore, the largest allowable value for N_C is 69.6% of N_R.

1.6. (*a*) Let cache access time be 1 and main memory access time be 20. Every instruction that is executed must be fetched from the cache, and an additional fetch from the main memory must be performed for 4% of these cache accesses. Therefore,

$$\text{Speedup factor} = \frac{1.0 \times 20}{(1.0 \times 1) + (0.04 \times 20)} = 11.1$$

(*b*)

$$\text{Speedup factor} = \frac{1.0 \times 20}{(1.0 \times 1) + (0.02 \times 20)} = 16.7$$